Vibrating Circular Membrane Simulation and Learning with Gaussian Markov Random Fields

Patrick Phillips

August 2019

Contents

1 Introduction

This report consists of six parts as laid out in the table of contents. In Section 2 I briefly derive the solution to the vibrating rectangular membrane with fixed boundary conditions. In Section 3 I describe the sequential Gaussian Markov Random Field (GMRF) regression algorithm for stationary (not time dependent) processes. In Section 4 I present an algorithm to formalize the combination of analytic PDE solutions with the sequential GMRF regression to create a learning process for time varying fields. In Section 5 I present a few figures and videos of my simulation results with various initial conditions. In Section 6 I have a few brief comments and highlight the findings of my report.

2 Analytic Solution of Vibrating Rectangular Membrance

The PDE that I am considering is the wave equation:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \tag{1}$$
$$\text{for } 0 < x < L, 0 < y < H$$

with boundary conditions

$$u(0, y, t) = 0 u(x, 0, t) = 0 \tag{2}$$
$$u(L, y, t) = 0 u(x, H, t) = 0$$

and initial conditions

$$u(x, y, 0) = \alpha(x, y) \tag{3}$$

$$\frac{\partial u}{\partial t}\ (x, y, 0) = \beta(x, y). \tag{4}$$

Applying the method of separation of variables by assuming a solution of the form

$$u(x, y, t) = h(t)\phi(x, y) \tag{5}$$

and plugging this into the original PDE along with separation constant λ results in

$$\frac{\mathrm{d}^2 h}{\mathrm{d}t^2} = -\lambda c^2 h. \tag{6}$$

As well as the eigenvalue problem

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = -\lambda \phi \tag{7}$$

with boundary conditions

$$\phi(0, y) = 0 \phi(x, 0) = 0 \tag{8}$$
$$\phi(L, y) = 0 \phi(x, H) = 0$$

Again, using separation of variables, this time for the eigenvalue problem, assuming that

$$\phi(x,y) = f(x)g(y) \tag{9}$$

Allows us to introduce a new separation variable γ, and yields the equations

$$\frac{\mathrm{d}^2 f}{\mathrm{d}x^2} + \gamma f = 0 \tag{10}$$

$$\frac{\mathrm{d}^2 g}{\mathrm{d}y^2} + (\lambda - \gamma) g = 0 \tag{11}$$

These two equations are just regular Sturm-Liouville eigenvalue problems, with eigenvalues γ and $(\lambda - \gamma)$ respectively. Using the boundary conditions to solve these yields that

$$\gamma_n = \left(\frac{n\pi}{L}\right)^2, n = 1, 2, 3, ... \tag{12}$$

With corresponding eigenfunctions

$$f_n(x) = \sin\frac{n\pi x}{L} \tag{13}$$

Now solving the other eigenvalue problem, we must use a second index, call it m, since there are an infinite number of distinct λ values for each γ_n. Solving 11 gives that

$$\lambda_{mn} - \gamma_n = \left(\frac{m\pi}{H}\right)^2, m = 1, 2, 3, ... \tag{14}$$

With corresponding eigenfunctions

$$g_{mn}(y) = \sin\frac{m\pi y}{H} \tag{15}$$

We can find λ explicitly by plugging in the known value of γ_n into equation 14. This gives

$$\lambda_{mn} = \left(\frac{n\pi}{L}\right)^2 + \left(\frac{m\pi}{H}\right)^2 \tag{16}$$

Now, putting together the solutions of $f_m(y)$ and $g_{mn}(y)$ to find ϕ_{mn} we get

$$\phi_{mn}(x,y) = \sin\frac{n\pi x}{L}\sin\frac{m\pi y}{H} \tag{17}$$

The solutions to $h(t)$ are

$$h_{mn}(t) = A_{mn}\cos c\sqrt{\lambda_{mn}}t + B_{mn}\sin c\sqrt{\lambda_{mn}}t \tag{18}$$

Putting everything together to find $u(x,y,t)$ as a linear combination of *all* possible solutions requires a sum over n and m, and yields

$$\begin{aligned} u(x,y,t) = &\sum_{m=1}^{\infty}\sum_{n=1}^{\infty} A_{mn}\sin\frac{n\pi x}{L}\sin\frac{m\pi y}{H}\cos c\sqrt{\lambda_{mn}}t \\ &+ \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} B_{mn}\sin\frac{n\pi x}{L}\sin\frac{m\pi y}{H}\sin c\sqrt{\lambda_{mn}}t \end{aligned} \tag{19}$$

To find A_{mn} and B_{mn}, we use the initial conditions. Applying the first initial of $u(x, y, 0) = \alpha(x, y)$ and multiplying by our eigenfunctions ϕ_{mn} and integrating over the domain, we find that

$$A_{mn} = \frac{4}{LH} \int_0^L \int_0^H \alpha(x, y)\phi_{mn}(x, y)dxdy \tag{20}$$

Similarly, we can use the initial condition for $\partial u/\partial t(x, y, 0) = \beta(x, y)$, to find our B_{mn} as

$$B_{mn} = \frac{4}{c\sqrt{\lambda_{mn}}LH} \int_0^L \int_0^H \beta(x, y)\phi_{mn}(x, y)dxdy \tag{21}$$

3 Gaussian Markov Random Field (GMRF)

The GMRF regression algorithm is designed for estimation of stationary (not time-dependent) spatial processes on a discrete grid of **G**aussian **R**andom variables (the G and R in GMRF). The **M**arkov assumption (the M in GMRF) is that any given vertex in the discrete grid is independent of all of the other vertices in the grid conditioned on its immediate neighbors. A generalization of the Markov process is a Markov blanket, or a CAR (j) process, which is where the *expectation* of a process value is defined through the next j adjacent graph vertices. This assumption enables the direct construction of a sparse precision matrix, and more efficient learning of the **F**ield (the F in GMRF).

Given a labeled graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V} = \{1,\ n\}$ and edges $\mathcal{E}$, a probabilistic graphical model η defines a GMRF, if the edges $\mathcal{E}$ are chosen such that there is no edge between node i and j, iff $\eta_i \perp \eta_j | \eta_{-ij}$, in which $-ij$ denotes the nodes adjacent to i and j. There is a theorem that states that the ij-th entry of the precision matrix, $\Lambda_{i,j}$ is zero iff $\eta_i \perp \eta_j | \eta_{-ij}$. Proof of this theorem can be found in [3].

In order to construct the GMRF, the spatial field $\mathcal{F}^* \subset R^d$ is discretized into a labeled undirected spatial graph with n^* vertex positions $\mathcal{S}^* = x_1^*, \ldots, x_n^*$, where x_i denotes the i-th field vertex position. The set of field locations $\mathcal{S}^*$ is extended to $\mathcal{S}$ with vertex positions $\mathcal{S} = x_1, \ldots x_n$ as depicted in Figure 1, to compensate boundary effects occurring due to the GMRF approximation. The field $\mathcal{F}^*$ can be extended to $\mathcal{F}$ and is then discretized into a regular grid $\mathcal{S}$ to enable the construction of a GMRF and to compensate for boundary effects.

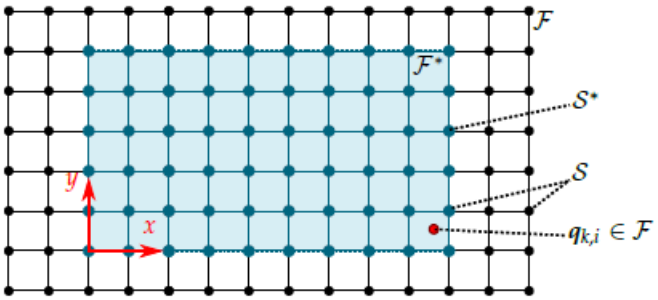


Figure 1: Field of the grid of random variables that the GRMF η is constructed on. Taken from my supervisor's paper [1]

Then on $\mathcal{S}$, a GMRF η can be constructed as $\eta \sim \mathcal{N}(0, \Sigma)$ is a GP on R^2 defined by the Matérn covariance function:

$$k_{\text{Matérn}}(x, x') = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left[\kappa \|x - x'\|\right])^{\nu} K_{\nu}(\kappa ||x - -x'\|) \tag{22}$$

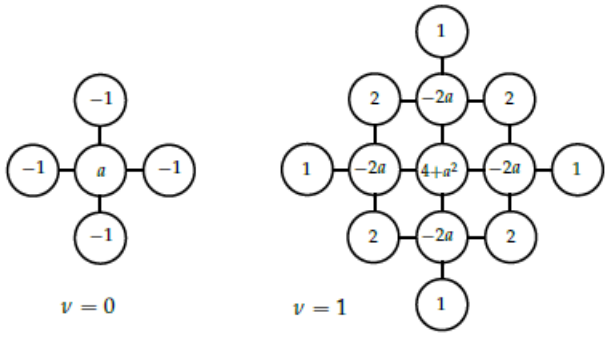


Figure 2: All of the elements of a column of the precision matrix associated with one random variable and its neighbor vertices on a regular two-dimensional GMRF lattice.

where $\|\cdot\|$ is the Euclidean distance in R^d and K_ν is the modified Bessel function of the second kind (I've looked into what the modified Bessel functions are defined as. I've also looked into why we want to use this covariance function, apparently defining the covariance in this way is important in proving that the GMRF constitutes a solution to a stochastic partial differential equation, for a discussion of this that I was not able to understand very well see [2]). The GMRF $\eta \sim \mathcal{N}(0, \Lambda^{-1})$ on the a regular two-dimensional grid approximates a Matérn GP for $\nu = 0$ if the full Gaussian conditionals are

$$\mathbb{E}\left(\eta|\boldsymbol{\eta}_{-ij}, \boldsymbol{\theta}\right) = \frac{1}{a}\left(\eta_{i-1,j} + \eta_{i+1,j} + \eta_{i,j-1} + \eta_{i,j+1}\right) = \frac{1}{a}\begin{pmatrix} \circ & \bullet & \circ \\ \bullet & \circ & \bullet \\ \circ & \bullet & \circ \end{pmatrix}, \ \mathrm{Pre}\left(\eta|\boldsymbol{\eta}_{-ij}, \boldsymbol{\theta}\right) = a\,\tau.$$

For the case of $\nu = 1$, the full Gaussian conditionals should be

$$\mathbb{E}\left(\eta|\boldsymbol{\eta}_{-ij}, \boldsymbol{\theta}\right) = \frac{1}{4+a^2}\begin{pmatrix} \circ\circ\circ\circ\circ & \circ\circ\circ\circ\circ & \circ\circ\bullet\circ\circ \\ \circ\circ\bullet\circ\circ & \circ\bullet\circ\bullet\circ & \circ\circ\circ\circ\circ \\ 2a\circ\bullet\circ\bullet\circ & -2\circ\circ\circ\circ\circ & -1\bullet\circ\circ\circ\bullet \\ \circ\circ\bullet\circ\circ & \circ\bullet\circ\bullet\circ & \circ\circ\circ\circ\circ \\ \circ\circ\circ\circ\circ & \circ\circ\circ\circ\circ & \circ\circ\bullet\circ\circ \end{pmatrix}, \ \mathrm{Pre}\left(\eta|\boldsymbol{\eta}_{-ij}, \boldsymbol{\theta}\right) = \left(4+a^2\right)\tau,$$

In each of these equations, $a = \kappa^2 + 4$ and τ adjusts the GMRF's signal variance independent of κ. The proof of these equations can be found in [4]. Figure 5 illustrates the correspondence between the spatial lattice locations and the values in each column of Λ, found using these equations.

In the GMRF algorithm, the full conditionals for the border vertices of the GMRF grid affect the estimation a lot. The commonly used boundary conditions are the Dirichlet, Neumann, and periodic boundary conditions.

The values of the field are represented by the latent variable $z_i = z(s_i) \in R$.

So the field belief on the lattice is denoted as $z = [z_1, \ldots, z_n]^{\mathrm{T}}$. These latent variables z_i are expressed as

$$z_i = (s_i, \beta) + \eta_i, \forall 1 \leq i \leq n \tag{23}$$

$$\mu(s_i, \beta) = m^{\mathrm{T}}\beta. \tag{24}$$

where, $m = [m_1\ (si),\ \cdots,\ m_p(s_i)]^{\mathrm{T}} \in \mathrm{R}^p$ denotes the regression function vector and the vector $\beta = [\beta_1, \ldots, \beta_p]^{\mathrm{T}}$ contains the unknown regression coefficients.

The GMRF $\eta \sim \mathcal{N}(0,\ \Lambda_{\eta|\theta)}$ is really just used to model the correlations of the field, where as the regression function vector models the mean values that are being learned at each point. I initialize the GMRF precision matrix $\Lambda^{-1}_{\eta|\theta}$ with the full conditionals as shown in figure 2 and given in equations as a diagonal matrix with large diagonal elements to represent high initial variance. The full precision matrix is written to include the regression vector coefficients as

$$\Lambda_{\overline{z}|\theta} = \begin{bmatrix} \Lambda_{\eta|\theta} & -\Lambda_{\eta|\theta}m \\ -m^{\mathrm{T}}\Lambda_{\eta|\theta} & m^{\mathrm{T}}\Lambda_{\eta|\theta}m + \mathbf{T} \end{bmatrix} \tag{25}$$

Where $\overline{z} = [z^{\mathrm{T}}, \beta^{\mathrm{T}}]^{\mathrm{T}} \in R^{n+p}$ with full probability distribution given as

$$\begin{aligned} p(\bar{\boldsymbol{z}}, \boldsymbol{\theta}) &= p(\boldsymbol{z}|\boldsymbol{\beta}, \boldsymbol{\theta})\, p(\boldsymbol{\beta}), \\ &\propto \exp\Big(-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{m\beta})^{\top}\boldsymbol{\Lambda}_{\boldsymbol{\eta}|\boldsymbol{\theta}}(\boldsymbol{z} - \boldsymbol{m\beta}) - \frac{1}{2}\boldsymbol{\beta}^{\top}\boldsymbol{T\beta}\Big), \\ &= \exp\big(-\frac{1}{2}\bar{\boldsymbol{z}}^{\top}\boldsymbol{\Lambda}_{\bar{\boldsymbol{z}}|\boldsymbol{\theta}}\bar{\boldsymbol{z}}\big), \end{aligned}$$

3.1 Shape Functions

The GMRF algorithm estimates the field on a discrete lattice $\{\mathcal{V},\ \mathcal{E}\}$ with vertices $\mathcal{V} = \{1,\ n\}$ and edges $\mathcal{E}$. To incorporate measurements taken off the grid, shape functions are introduced. The set of continuous field locations is discretized into a finite subset of n spatial input locations $\mathcal{S} = \{s_1,\ s_n\}$. The lattice $\mathcal{S}$ consists of a finite number of sub-domains $\mathcal{S}_{e,i}$, where each is enclosed by four vertices $\overline{s}_i$, with $i = 1, 2, 3, 4$. For the ease of illustration, $\mathcal{S}$ is chosen as regular lattice with edges each of length a and b respectively, as depicted in Figure 3. The field value at position q can be approximated through a sum of weighted shape functions (basically a weighted mean of the vertices enclosing the measurement location).

A local coordinate system $\mathcal{K}_e(x_e, y_e)$ is defined on $\mathcal{F}_e$. The origin of $\mathcal{K}_e(x_e, y_e)$ lies in the center of the element. The corresponding coordinate axis are orthogonal to each other and parallel to the respective element edges. The shape functions are defined as

$$\phi^e_1 = \frac{1}{ab}(x_e - \frac{a}{2})\ (y_e - \frac{b}{2})\ ,\ \phi^e_2 = -\frac{1}{ab}(x_e + \frac{a}{2})\ (y_e - \frac{b}{2})\ ,$$

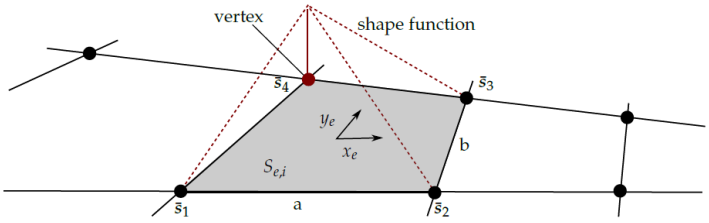


Figure 3: Depiction of the weighted mean of the shape functions.

$$\phi_3^e = \frac{1}{ab}(x_e + \frac{a}{2})\ (y_e + \frac{b}{2})\ ,\ \phi_4^e = -\frac{1}{ab}\ (x_e - \frac{a}{2})(y_e + \frac{b}{2})$$

And also for brevity, define the vector $\Phi_{1:k}$ as $\Phi_{1:k} = [\Phi_1^{\mathrm{T}}, \ldots, \Phi_k^{\mathrm{T}}]^{\mathrm{T}}$

3.2 Sequential GMRF Regression Algorithm

In the regression algorithm, the canonical mean b_k defined as $b_k = \Lambda_{k|\theta}\mu_{k|\theta}$ is used instead of the normal predictive mean because it allows for easier updates without computing the covariance matrix which is a computationally expensive operation compared to the sparse matrix operations of the precision matrix.

The update rules for sequential GMRF regression come from the formula of conditioning of a multivariate Gaussian distribution which can be found in [3]. The update rules are

$$\begin{aligned} p(\bar{\boldsymbol{z}}|\boldsymbol{\theta}, \boldsymbol{y}_{1:k}) &= \mathcal{N}(\boldsymbol{\Lambda}_{k|\boldsymbol{\theta}}^{-1}\boldsymbol{b}_k, \boldsymbol{\Lambda}_{k|\boldsymbol{\theta}}^{-1}), \\ \boldsymbol{\Lambda}_{k|\boldsymbol{\theta}} &= \boldsymbol{\Lambda}_{\bar{z}|\boldsymbol{\theta}} + \frac{1}{\sigma_{\tilde{y}}^2}\boldsymbol{\Phi}_{1:k}^{\top}\boldsymbol{\Phi}_{1:k} = \boldsymbol{\Lambda}_{k-1|\boldsymbol{\theta}} + \frac{1}{\sigma_{\tilde{y}}^2}\boldsymbol{\Phi}_k^{\top}\boldsymbol{\Phi}_k, \\ \boldsymbol{b}_k &= \frac{1}{\sigma_{\tilde{y}}^2}\boldsymbol{\Lambda}_{k|\boldsymbol{\theta}}\boldsymbol{\Lambda}_{k|\boldsymbol{\theta}}^{-1}\boldsymbol{\Phi}_{1:k}^{\top}\boldsymbol{y}_{1:k} = \boldsymbol{b}_{k-1} + \frac{1}{\sigma_{\tilde{y}}^2}\boldsymbol{\Phi}_k^{\top}\boldsymbol{y}_k, \end{aligned}$$

The predictive mean can simply be reconstructed from the canonical mean as $\mu_k = \Lambda_k^{-1}b_k$. The predictive variance is given as $\mathrm{diag}(\Sigma_k)$, and the derivation of the update rule for this can be found by using the Sherman-Morrison formula [3] [1], and is seen in the Sequential GMRF Regression algorithm below.

Algorithm 1: Sequential GMRF Regression

INPUT: Canonical mean $b_0 = 0$, Precision matrix $\Lambda_0 = \Lambda_{\overline{z}}$, Hyperparameter vector θ, Extended field grid $\mathcal{S}$, Regression function vector m, Measurement variance σ_y^2

begin

compute $\text{diag}(\Sigma_0) = \text{diag}\ (\Lambda_{\overline{z}}^{-1})$;

for $k \in Z > 0$ **do**

get agent location x_k and measurement y_k;

compute $\Phi_k(x_k, \mathcal{S})$;

$b_k = b_{k-1} + \frac{1}{\sigma_y^2}\Phi_k^{\mathrm{T}} y_k$;

$\Lambda_k = \Lambda_{k-1} + \Sigma_{l=1}^{N} \frac{1}{\sigma_y^2}\Phi_{k,l}^{\mathrm{T}}\Phi_{k,l}$;

$h_k = \Lambda_{k-1}^{-1}\Phi_k^{\mathrm{T}}$;

$\mu_k = \Lambda_k^{-1} b_k$;

$\text{diag}(\Sigma_k) = \text{diag}(\Sigma_{k-1}) - \Sigma_{l=1}^{N} \frac{h_k^T \cdot h_k}{\sigma_y^2 + \Phi_{k,l} h_{k,l}}$;

end

end

OUTPUT: The GMRF mean μ_k and GMRF variance which is $\text{diag}(\Sigma_k)$

An extension of the model, is described in Xu et al. to incorporate a hyperparameter estimation [4]. This would look like defining a maximum a posteriori distribution $p(\theta|y) \sim p(y|\theta)p(\theta)$ with $p(\theta)$ being a uniform prior distribution over a discrete set of possible hyperparameters (the notation in 4 uses $\pi(\theta)$). Another extension of the model, incorporated in the Xu et al's algorithm shown in Figure 4, takes account of N agents performing measurements simultaneously. My implementation was based off of Xu et al's as well as of course my supervisor's Daniel Deucker [1], and is for only a single agent, but incorporates the hyperparameter estimation. On the next page is the GMRF regression algorithm as written Xu et Al, apologies for the different notation, if there are terms that can't be mapped to the ones used in this report, or for a complete explanation of this algorithm, see [4].

TABLE I

SEQUENTIAL BAYESIAN PREDICTIVE ALGORITHM.

Input: (1) prior distribution of $\theta \in \Theta$, *i.e.*, $\pi(\theta)$
(2) spatial sites $\mathcal{S}_* := \{s_1, \cdots, s_{n_*}\}$
(3) extended sites $\mathcal{S} := \{s_1, \cdots, s_n\}$
(4) regression function $f(\cdot)$
(5) number of regression coefficients p

Output: (1) predictive mean $\mu_{x_i|y_{1:t}}$
(2) predictive variance $\sigma^2_{x_i|y_{1:t}}$

Initialization:

1: initialize $b = 0 \in \mathbb{R}^{n+p}$, $u = 0 \in \mathbb{R}^{n+p}$, $c = 0 \in \mathbb{R}$
2: **for** $\theta \in \Theta$ **do**
3: initialize $Q_\theta \in \mathbb{R}^{(n+p)\times(n+p)}$, $g_\theta = 0 \in \mathbb{R}$
4: compute $\text{diag}(Q_\theta^{-1}) \in \mathbb{R}^{n+p}$
5: **end for**

At time $t \in \mathbb{Z}_{>0}$, do:

1: **for** $1 \le i \le N$ **do**
2: obtain new observations $y_{t,i}$ collected at current locations $q_{t,i}$
3: find the index k corresponding to $q_{t,i}$, and set $(u)_k = 1$
4: update $(b)_k = (b)_k + y_{t,i}/\sigma_w^2$
5: update $c = c - y_{t,i}^2/(2\sigma_w^2)$
6: **for** $\theta \in \Theta$ **do**
7: compute $h_\theta = Q_\theta^{-1} u$
8: update $\text{diag}(Q_\theta^{-1}) = \text{diag}(Q_\theta^{-1}) - \frac{h_\theta \circ h_\theta}{\sigma_w^2 + u^T h_\theta}$
9: update Q_θ via $(Q_\theta)_{kk} = (Q_\theta)_{kk} + 1/\sigma_w^2$
10: update $g_\theta = g_\theta - \frac{1}{2}\log(1 + \frac{1}{\sigma_w^2} u^T h_\theta)$
11: **end for**
12: **end for**
13: **for** $\theta \in \Theta$ **do**
14: compute $\mu_\theta = Q_\theta^{-1} b$
15: compute the likelihood via $\log \pi(y_{1:t}|\theta) = c + g_\theta + \frac{1}{2} b^T \mu_\theta$
16: **end for**
17: compute the posterior distribution via $\pi(\theta|y_{1:t}) \propto \pi(y_{1:t}|\theta)\pi(\theta)$
18: compute the predictive mean via $\mu_{x_i|y_{1:t}} = \sum_\ell (\mu_{\theta_\ell})_i \pi(\theta_\ell|y_{1:t})$
19: compute the predictive variance via $\sigma^2_{x_i|y_{1:t}} = \sum_\ell \left((\text{diag}(Q_{\theta_\ell}))_i + ((\mu_{\theta_\ell})_i - \mu_{x_i|y_{1:t}})^2 \right) \pi(\theta_\ell|y_{1:t})$

Figure 4: Sequential GMRF regression algorithm by Xu et Al [4]

4 Updating the Field in Time

To update the field in time, we solve the PDE. In addition to knowing the governing PDE, I also need boundary conditions, and initial conditions. I assume that I know the boundary conditions (an assumption I discuss in the conclusion). For my first initial conditions I use the mean's, μ's, of the GMRF variables as the initial value for the PDE. I use the partial derivative with respect to time of my previous solution to the PDE I found in the last time step as my second initial condition. Explicitly, these initial conditions are

$$\begin{aligned} \alpha &= u_i(x, y, 0) = \mu_{i-1}(x, y) \\ \beta &= \frac{\partial u_i}{\partial t}\ (x, y, 0) = \frac{\partial u_{i-1}}{\partial t}\ (x, y, \mathrm{dt}) \end{aligned} \tag{26}$$

where u_i represents the i-th PDE solution, and μ_{i-1} represents the predictive mean from the sequential GMRF regression algorithm. Note that $\frac{\partial u_{i-1}}{\partial t}$ is evaluated at time dt since this much time has passed since the solution was found.

A problem arises in using the first initial condition; for $\mu(x, y)$; we have discrete values at different points across the vibrating membrane. Looking back at how we solved the PDE analytically, we use this initial condition in solving for A_{mn} as seen in equation 20. So we can simply change the integral to a summation, and use our discrete set of μ values to accurately approximate this integral and solve for our A_{mn}, using $dxdy = dA = \Delta A$ where ΔA is a small amount of area, which as seen in Figure 5, should just be the distance between vertices squared, ℓ^2. In implementation, scipy.integrate.simps() is used which uses Simpson's numerical integral rule. This rule is similar to a Riemann sum which uses rectangles to approximate the integral, but instead Simpson's rule uses parabolas to approximate the integral.

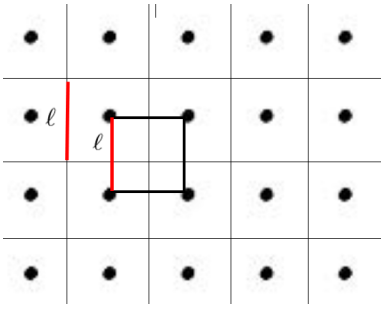


Figure 5: Showing the grid length of the discrete field of the GMRF vertices

Note that since we assume we know the BC's, "solving the PDE" really only consists of finding these constants $A_{m,n}$ as well as $B_{m,n}$. During implementation, it was discovered that using scipy.integrate.simps() twice for the double integral over a discrete grid was much faster than alternatives such as perform-

ing numerical integration of a defined function using other methods, so this was also done to find the $B_{m,n}$.

In order to use the previous solution to the wave equations as an approximate initial condition for the current solution, we must first find the partial derivative with respect to time of $u(x, y, t)$. From equation 19 this is just

$$\begin{aligned}\frac{\partial u}{\partial t}\ (x,y,t) = &\frac{1}{c\sqrt{\lambda_{mn}}} \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} B_{mn} \sin\frac{n\pi x}{L} \sin\frac{m\pi y}{H} \cos c\sqrt{\lambda_{mn}}t \\ &-\frac{1}{c\sqrt{\lambda_{mn}}} \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} A_{mn} \sin\frac{n\pi x}{L} \sin\frac{m\pi y}{H} \sin c\sqrt{\lambda_{mn}}t\end{aligned} \tag{27}$$

The algorithm for incorporating the time updates by solving the PDE is formalized below in Algorithm 2.

Algorithm 2: GMRF regression with PDE time-stepping updates

INPUT: Canonical mean $b_0 = 0$, Precision matrix $\Lambda_0 = \Lambda_{\overline{z}}$, Hyperparameter vector θ, Extended field grid $\mathcal{S}$, Regression function vector m, Measurement variance σ_y^2

begin

compute $\text{diag}(\Sigma_0) = \text{diag}\ (\Lambda_{\overline{z}}^{-1})$;

for $k \in Z > 0$ **do**

obtain α as $\mu_{i-1}(x, y)$;

obtain β as $\frac{\partial u_{i-1}}{\partial t}$;

find $A_{m,n}$ and $B_{m,n}$ with numerical integration;

update $\mu_k = u(x, y, dt)$ at all grid points;

update canonical mean as $b_k = \Lambda_k \mu_k$;

get agent location x_k and measurement y_k;

compute $\Phi_k(x_k, \mathcal{S})$;

$b_k = b_{k-1} + \frac{1}{\sigma_y^2}\Phi_k^{\mathrm{T}} y_k$;

$\Lambda_k = \Lambda_{k-1} + \Sigma_{l=1}^{N} \frac{1}{\sigma_y^2}\Phi_{k,l}^{\mathrm{T}}\Phi_{k,l}$;

$h_k = \Lambda_{k-1}^{-1}\Phi_k^{\mathrm{T}}$;

$\mu_k = \Lambda_k^{-1} b_k$;

$\text{diag}(\Sigma_k) = \text{diag}(\Sigma_{k-1}) - \Sigma_{l=1}^{N} \frac{h_k^T \cdot h_k}{\sigma_y^2 + \Phi_{k,l} h_{k,l}}$;

end

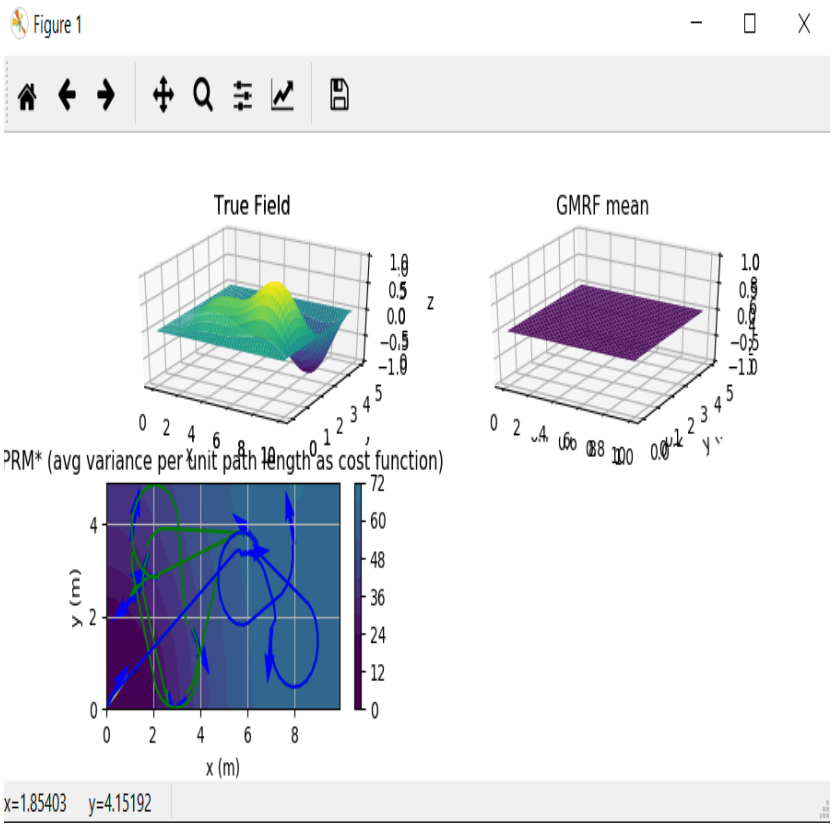
end

OUTPUT: The GMRF mean μ_k and GMRF variance which is $\text{diag}(\Sigma_k)$

5 Simulation and Results

The implementation of Algorithm 2 is done in the *gmrf-bayese-update()* section of *gp-scripts.py*. The *main.py* file is called to run the process. Various parameters can be set in the *Config.py* file. Plots comparing the various simulations are generated from the *plot-data.py*, and live plots of the simulation are generated from the *plot-scripts.py*. Code can be found on my Github (and should also be attached). The **data** folder stores data from the various runs. The **control algorithms** folder includes various path planning algorithms that have been set up with minimizing predictive variance as the objective function (I have a separate report I wrote on these control algorithms if you're interested).

There are three videos, with three sets of different initial conditions included in this report. Hopefully the pdf graphics don't break, but I will send the videos separately in case they do. Of course feel free to try out different initial conditions, they are set at the top of the 'vibrating-membrane-main.py' file as functions 'alpha' and 'beta'. The video shows the analytic field in the top left, labelled "True Field", the GMRF belief in the top right labelled "GMRF mean", and the path that the agent is taking plotted on top of a contour plot of the variance (the diagonal of the covariance matrix). The title of this plot just reflects the control algorithm that is being used.

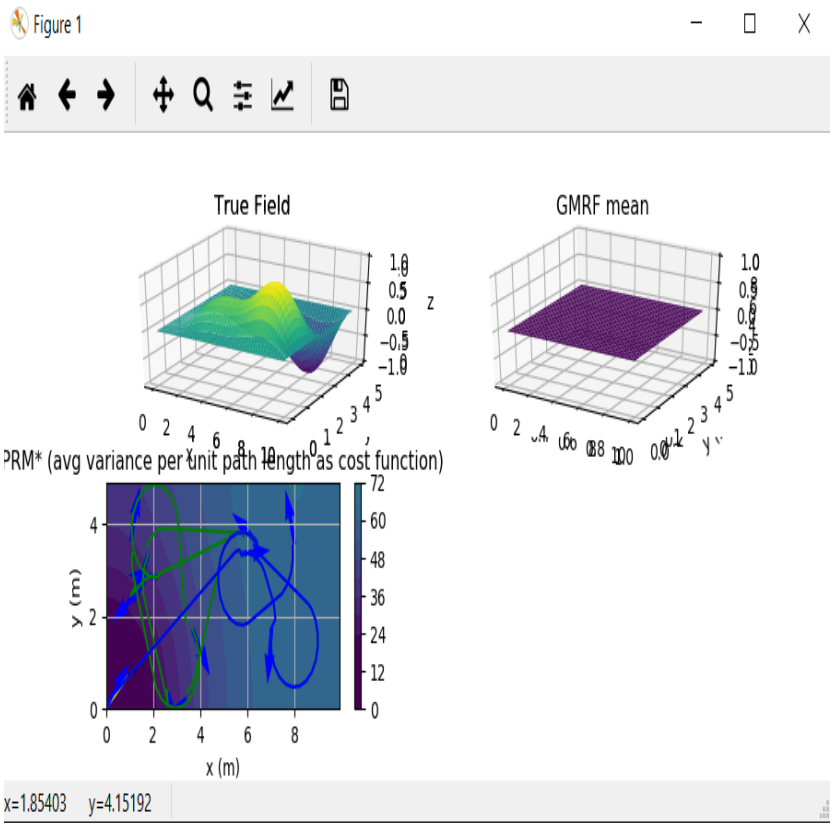


Video 1: Video of the algorithm described in this report combining sequential GMRF regression with PDE updates.

This first video uses initial conditions

$$\alpha = (x - L)(y - H) * x * y$$
$$\beta = \sin x \cos y$$

These initial conditions clearly satisfy the BC's as α is zero at x=0, y=0, x=L, or y=L.

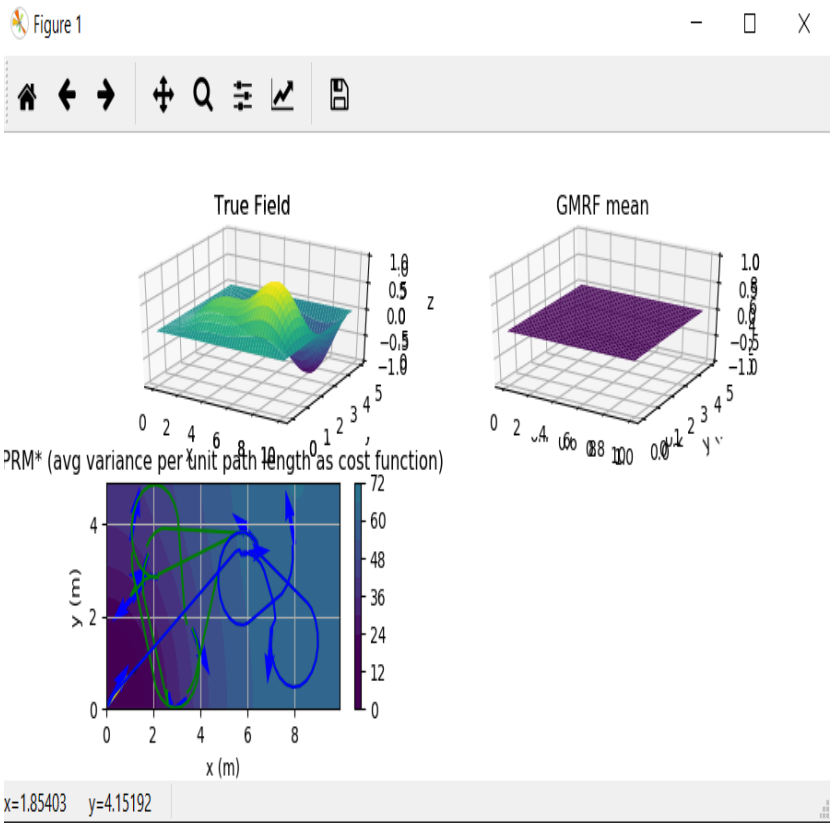


Video 2: Video of the algorithm described in this report combining sequential GMRF regression with PDE updates.

The second video above uses the initial conditions

$$\alpha = \sin\frac{\pi x}{L}\sin\frac{\pi y}{H}$$
$$\beta = 0$$

Where the α here can ben seen to be just the first eigenfunction of the analytic field.



Video 3: Video of the algorithm described in this report combining sequential GMRF regression with PDE updates. Initial conditions for this simulation are described in section 5

The last video shown above uses the initial conditions

$$\alpha = \sin x \cos y$$
$$\beta = 0$$

The α here clearly does **not** satisfy the boundary conditions. I was interested to see how the true field behaved in this case, given a function that did not satisfy the boundary condition as an IC. It acted as I expected, and takes on a shape that is similar to the true shape of $\sin x \cos y$, but satisfies the BC's.

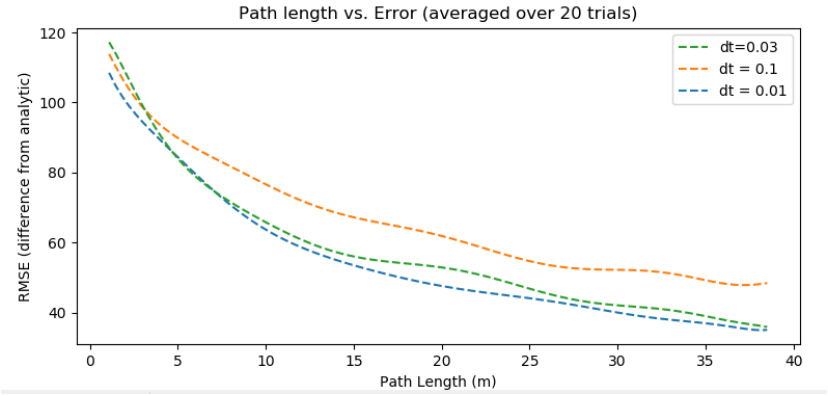


Figure 6: Plot of error vs path length. Path length of the agent is used, because not all trials ran for the same amount of time, but they all ran for the same path length. The error is measured as the average difference between the grid point mean in the GMRF and the corresponding point in the analytic field.

From the first video and first set of initial conditions, data was collected on the difference between the true field and GMRF approximated field. This data is plotted in Figure 6. On the x-axes is the path length of the path the agent has taken, on the y-axes is the root mean squared error (RMSE). This is a measure of the average difference between the analytic field and the true field. As can be seen the error clearly decreases with path length (and thus increasing time and number of measurements). The error also decreases with simulations that use smaller time steps 'dt'.

6 Comments and Conclusion

One thing I noticed while writing the report and adding videos is that the boundary conditions are not always satisfied by the belief field. This is peculiar because the way the algorithm works is by assuming the boundary conditions when making a time update. However, this makes sense with the way the learning algorithm is implemented, because the information from the measurement is taken in after the PDE's update in time, and the GMRF algorithm of course has no reason to obey the boundary conditions. It was verified that the solution to the PDE did indeed satisfy the BC's, as it must because of the eigenfunctions $\phi_{m,n}$ given in equation 17. Although it still seems peculiar that passing into the PDE initial conditions that don't satisfy the boundary conditions still results in a field that *do* satisfy the initial conditions. Although examining how the constants $A_{m,n}$ and $B_{m,n}$ are solved for in equations 20 and 21, I would not expect that any numerical problems would arise.

7 Conclusion

One thing I'm interested to see is how using a combination of sequential GMRF regression and PDE solution compares to the Spacetime Kalman Filter (STKF) algorithm that was briefly mentioned. I plan on implementing a version of the STKF and comparing the RMSE of each to see which is superior in matching the analytic solution. Of course, the added information in using the boundary conditions in the GMRF algorithm is clearly a bit of an unfair advantage. Thus to eliminate this difference, and for making this method for universally applicable, I plan to investigate ways to not use an assumption over the boundary conditions when solving the PDE, such as perhaps using the Fourier Transform solution.

References

[1] Daniel Andre Duecker, Andreas Rene Geist, Edwin Kreuzer, and Eugen Solowjow. Learning environmental field exploration with computationally constrained underwater robots: Gaussian processes meet stochastic optimal control. *Sensors*, 19(9):2094, 2019.

[2] Finn Lindgren, Håvard Rue, and Johan Lindström. An explicit link between gaussian fields and gaussian markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498, 2011.

[3] Havard Rue and Leonhard Held. *Gaussian Markov random fields: theory and applications*. Chapman and Hall/CRC, 2005.

[4] Yunfei Xu, Jongeun Choi, Sarat Dass, and Tapabrata Maiti. Efficient bayesian spatial prediction with mobile sensor networks using gaussian markov random fields. *Automatica*, 49(12):3520–3530, 2013.